Universidade Estadual de Campinas – Instituto de Computação
Segundo Semestre de 2015
MC202 – Estrutura de Dados – Turmas G e H
Professor Lehilton Pedrosa

Segunda lista de exercícios

1. Desenhe todas as árvores binárias de busca para os quatro elementos A, B, C e D.

2. Quais é o número mínimo e máximo de nós para uma árvore de altura h?

3. Escreva uma função **recursiva** para contar o número de nós pertencentes ao nível l de uma árvore.

4. Suponha que o percorrimento em largura de uma árvore binária foi $0,1,2,3,4,5,6,7$ e o percorrimento em profundidade pré-ordem foi $0,1,3,2,4,5,6,7$. Não é possível reconstruir a árvore original só com essas informações. Desenhe duas árvores de alturas 4 e 5 com os mesmos percorrimentos. (**corrigido!** Na versão anterior a pré-ordem estava errada, desculpem.)

5. Implemente a inserção e remoção recursiva de uma árvore de busca.

6. Qual o resultado do percorrimento pós-ordem após a inserção dos elementos $17,11,15,5$, 10, 8, 11 e remoção do 15.

7. Uma *árvore de estatística de ordem* é uma variante das árvores binárias de busca que contém as seguintes operações adicionais:

- *selecionar(i)*: encontra o i-ésimo elemento da árvore;
- *ordem(x)*: encontra a ordem em que se encontra um elemento de chave x.

Implemente um TAD *árvore de estatística de ordem*. As operações acima devem ter tempo de execução $O(h)$, em que h é a altura da árvore.

8. Partindo de uma árvore AVL vazia, realize a inserção da seguinte sequência de chaves:

99, 44, 71, 80, 74, 63, 59, 120, 98, 150.

Redesenhe a árvore a cada inserção. Indique para cada rotação feita, o nome da rotação e o nó desregulado. Indique as árvores resultantes da exclusão dos nós 59 e 63.

9. A altura de uma árvore AVL está relacionada com a sequência de Fibonacci.

a) Seja $n(h)$ o número mínimo de nós em uma árvore de altura h. Encontre uma fórmula de recorrência para $n(h)$.

b) Demonstre que $n(h) = f(h+2) - 1$, onde $f(i)$ é o i-ésimo elemento da sequência de Fibonacci 1, 1, 2, 3, 5, ... Dica: calcule $n(h)+1$ para $h = 0,1,2,3,4$

c) Utilizando o fato

$$f(x) = O\left(\left(\frac{1+\sqrt{5}}{2}\right)^x\right),$$

mostre a altura de uma árvore AVL (máxima) com n nós é no máximo $1,441\log_2 n + C$, onde C é uma constante.

10. Disserte:

a) Sobre árvores de busca, entre outros pontos, você deve abordar: a importância dessas árvores e qual o problema elas resolvem; o que é e porque queremos árvores balanceadas; defina árvores AVL, vermelho-preto e de afunilamento; compare a dificuldade de implementação de cada árvore.

b) Considere a seguinte afirmação: *"Se em um programa, a quantidade de vezes que realizamos operações sobre um conjunto (inserir, remover) for bastante reduzida, comparada com o número de*

acessos, então é vantajoso usar uma árvore-B na memória a usar uma árvore binária AVL, porque a altura da árvore binária é bem maior que a de uma árvore de ordem $b = 1000$, isso é, $\log_2 n >> \log_b n$.*"* Concorde ou discorde da afirmação. Escreva um pequeno parágrafo que justifique sua posição.

11. Suponha que uma árvore-B tenha ordem $b = 2m$. Responda:

a) Quando um nó precisa ser dividido?

b) Qual o número mínimo de nós no nível $l \geq 2$.

c) Suponha que uma inserção tenha acabado de aumentar a altura da árvore-B. A raiz antiga foi dividida em duas. Quais opções são corretas:

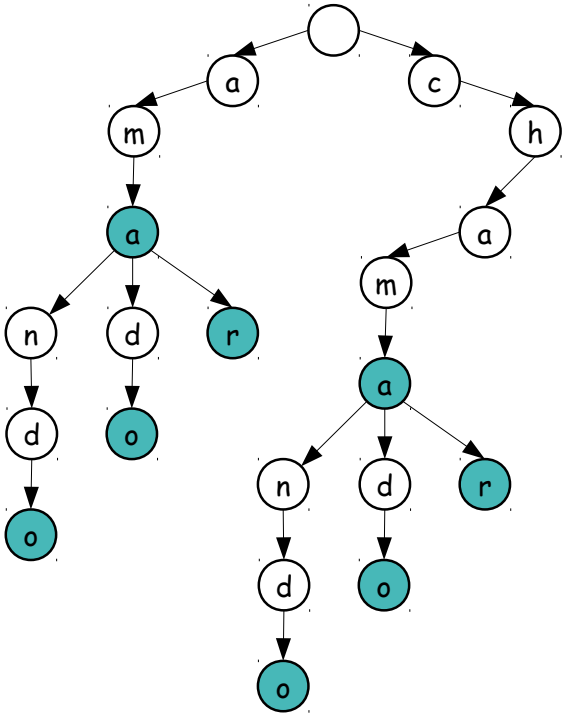
- A nova raiz tem um filho com só um nó.
- O número de folhas da subárvore esquerda é igual ao da direita.
- Se o número de elementos da subárvore esquerda é E e o do da direita é D, então $D/m \leq E \leq m \cdot D$.

12. Encontre uma árvore-B de ordem três inserindo a sequência de números a seguir: 91, 8, 43, 94, 78, 74, 74, 92, 98, 59.

13. Encontre uma árvore vermelho-preta inserindo a sequência de números a seguir: 91, 8, 43, 94, 78, 74, 74, 92, 98, 59. Compare com a árvore-B.

14. Você recebe k listas de números já ordenadas, com um total de m elementos. Escreva um programa para criar uma lista ordenada com todos os m elementos. A complexidade do algoritmo deve ser $O(m \log k)$.

15. Às vezes só estamos interessados nas chaves existentes de uma árvore de prefixo (isso é, não existem dados associados a cada um dos nós). Considere a seguinte árvore de prefixo e note que existe algumas redundâncias entre eles. Como você pode aproveitar esse fato? Desenhe uma estrutura de dados que usa menos nós do que essa (não precisa ser árvore). Você consegue dizer se a estrutura que você desenhou tem o menor número de nós?



16. Em uma máquina fictícia, cada unidade de memória é um *trit*, isso é, ao invés de armazenarmos sequências de 0 ou 1, podemos armazenar sequências de 0, 1 ou 2. Huffmão, um aluno de estrutura de dados, propôs o seguinte algoritmo para encontrar uma árvore de codificação de trits que gera a sequência de menor tamanho.

1. Criar um nó para cada símbolo com sua frequência;
2. Inserir os nós numa fila de prioridade (usando a frequência como prioridade);
3. Enquanto houver mais de um elemento na fila:
   (a) Desenfileire 3 elementos se houver, ou senão desenfileire os 2 últimos;
   (b) Crie um novo nó com a soma das frequências;
   (c) Adicione os nós desenfileirados nos índices como filhos do novo nó (nos índices 0, 1 e 2);
   (d) Insira o novo nó na fila
4. Desenfileire o último nó da fila e devolva

a) Experimente este algoritmo com algum conjunto de palavras de sua preferência. Desenhe a árvore gerada.

b) A codificação de Huffmão é a mínima? Se não for, dê um exemplo que ela não é mínima.

c) (desafio) Se a codificação gerada for mínima, então mostre que o algoritmo está correto. Se não for, então corrija ou crie um algoritmo para encontrar a codificação de tamanho mínimo e mostre que ele está correto.